EXHIBIT 2

FILED UNDER SEAL

CLEMENT SETH ROBERTS (STATE BAR NO. 209203)
croberts@orrick.com
BAS DE BLANK (STATE BAR NO. 191487)
basdeblank@orrick.com
ALYSSA CARIDIS (STATE BAR NO. 260103)
acaridis@orrick.com
EVAN D. BREWER (STATE BAR NO. 304411)
ebrewer@orrick.com
ORRICK, HERRINGTON & SUTCLIFFE LLP
The Orrick Building
405 Howard Street
San Francisco, CA 94105-2669
Telephone: +1 415 773 5700
Facsimile: +1 415 773 5759

SEAN M. SULLIVAN (*pro hac vice*)
sullivan@ls3ip.com
COLE RICHTER (*pro hac vice*)
richter@ls3ip.com
LEE SULLIVAN SHEA & SMITH LLP
656 W Randolph St., Floor 5W
Chicago, IL 60661
Telephone: +1 312 754 0002
Facsimile: +1 312 754 0003

Attorneys for Defendant Sonos, Inc.

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

GOOGLE LLC,

Plaintiff and Counter-defendant,

v.

SONOS, INC.,

Defendant and Counter-claimant.

Case No. 3:20-cv-06754-WHA
Related to Case No. 3:21-cv-07559-WHA

FIRST SUPPLEMENTAL REPLY EXPERT REPORT OF DR. KEVIN C. ALMEROTH

Rebuttal Report:

```
base::flat_set<std::string> group_uuids({virtual_group_uuid_});
  for (const auto& g : local_groups) {
    group_uuids.insert(g.uuid);
    auto it = groups_.find(g.uuid);
    if (it == groups_.end()) {
      StopCurrentApp();
      AddGroup(g);
    } else if (it->second->Reconfigure(g)) {
      SaveGroupConfig(g);
    } else {
      continue;
    }
    groups_changed = true;
```

Schonfeld Rebuttal Report at ¶ 59 (citing SC-GOOG-SONOSNDCA-001637 – 38); *see also* 1/25/2023 K. MacKay Rough Dep. Tr. at 9:19-11:4, 23:13-16 (Google's corporate designee testifying that the only change made to the RefreshDeviceGroups() function relative to prior versions of the source code was the addition of the StopCurrentApp() function and also confirming that there were no changes made to the AddGroup() function called by the RefreshDeviceGroups() function relative to prior versions of the source code); Sonos Dep. Ex. 1320-1321.

93. In this respect, my understanding of how this additional call to the MultizoneManager::StopCurrentApp() function impacts the functionality of an Accused Google Player installed with newly-released firmware version 1.56.324896 is that, if such an Accused Google Player is running a particular receiver app at the time that it receives "join_group" message indicating that the Accused Google Player has been added to a new speaker group (e.g., the YouTube Music receiver app), the MultizoneManager::StopCurrentApp() function will cause the Accused Google Player to stop that particular receiver app. In this respect, if the particular receiver app being run by the Accused Google Player is currently causing the Accused Google Player to engage in active playback, then the MultizoneManager::StopCurrentApp() function will cause the Accused Google Player to stop that active playback, whereas if the receiver app being run by the Accused Google Player is not currently causing the Accused Google Player to engage in active playback, then the MultizoneManager::StopCurrentApp() function will not impact the playback state of the Accused Google Player. However, in either case, the additional call to the MultizoneManager::StopCurrentApp() function does not cause an Accused Google Player

operating in a standalone mode to transition into a grouped mode in which it operates in accordance with the new speaker group. Rather, the MultizoneManager::StopCurrentApp() function causes an Accused Google Player operating in a standalone mode to stop its currently-running receiver app (to the extent that there is a receiver app currently running) while the new speaker group remains in an unlaunched state and the Accused Google Player remains in standalone mode.

94. My understanding of the MultizoneManager::StopCurrentApp() function has been confirmed by the testimony of Google's corporate designee, Mr. Kenneth MacKay. *See, e.g.,* 1/25/2023 K. MacKay Rough Dep. Tr. at 15:22-21:8 (testifying about the portion of the RefreshDeviceGroups() function where the StopCurrentApp() function is called), 23:17-33:9 (describing the operation of the StopCurrentApp() function), 33:10-13 (confirming that the StopCurrentApp() function will not "perform any checking of group state as part of stopping the app"), 33:14-34:1 (confirming that there is "no group information" that is passed into the StopCurrentApp() function), 34:2-9 (confirming that the StopCurrentApp() function does not cause any speaker group to be launched); 41:22-48:2 (describing the operation of the StopCurrentApp() function and how it differs from the StopApp() and StopPlayback() functions), 48:12-55:9, 56:16-62:14 (describing the operation of the StopCurrentApp() function in different scenarios for creating and modifying speaker groups).[7]

95. Moreover, the other functions in the source code path for receiving and handling a "join_group" message for a new speaker group do not appear to meaningfully differ from the functions included in the foregoing source code path that was already found to infringe the "continuing to operate in the standalone mode" limitation of Asserted Claim 1 of the '885 Patent.

[7] Mr. MacKay also testified that there may be scenarios where the StopCurrentApp() function will not stop a receiver app on an Accused Google Player installed with newly-released firmware version 1.56.324896, such as a scenario where the Accused Google Player is running a "non-visible app," as well as scenarios where a stopped current app is immediately replaced by another app that was "pending" and/or "preloaded" in the background at the time that the StopCurrentApp() function executes. *See* 1/25/2023 K. MacKay Rough Dep. Tr. at 27:7-28:1, 31:6-32:1, 43:15-45:6. These scenarios provide further support for my opinions, because even under Dr. Schonfeld's apparent theory that no longer running an app amounts to leaving "standalone mode," there will be scenarios where an Accused Google Player has at least one app running (or at least loaded) after executing the StopCurrentApp() function.

> is added to a group that is not playing back music will stop playback when added to that group.

Schonfeld Rebuttal Report at ¶ 47. However, this statement is flawed for several reasons.

98. First, Dr. Schonfeld's suggestion that "speakers added to the group *no longer continue their previous activity* and instead either play back music (if that is what the group was doing) or stop playback to match the group's state of stopped playback" is incomplete and inaccurate. As set forth above in Section IX.A, there are a number of scenarios where Accused Google Players installed with newly-released firmware version 1.56.324896 "continue their previous activity" after being added to a speaker group. For instance, in any scenario where an Accused Google Player installed with newly-released firmware version 1.56.324896 is operating in a standalone mode and is not engaging in active playback at the time when it is added to a new speaker group, that Accused Google Player will "continue [its] previous activity" after being added to the speaker group by continuing to operate in standalone mode and continuing not to engage in active playback. Likewise, in any scenario where an Accused Google Player installed with newly-released firmware version 1.56.324896 is operating in a standalone mode and is not engaging in active playback at the time when it is added to a pre-existing speaker group that is unlaunched, that Accused Google Player will "continue [its] previous activity" after being added to the pre-existing speaker group by continuing to operate in standalone mode and continuing not to engage in active playback.

99. Second, for similar reasons, Dr. Schonfeld's suggestion that "[s]peakers added to a speaker group do not continue with their previous playback or *non-playback state* when added to a group" is incomplete and inaccurate. Again, as set forth above in Section IX.A, there are a number of scenarios where Accused Google Players installed with newly-released firmware version 1.56.324896 "continue with their previous . . . *non-playback state*" after being added to a speaker group, including but not limited to the scenarios mentioned in the preceding paragraph.

100. Third, Dr. Schonfeld's suggestion that when an Accused Google Player installed with newly-released firmware version 1.56.324896 is added to a new speaker group, the Accused Google Player "stop[s] playback to match the group's state of stopped playback" is an inaccurate

and misleading characterization of the functionality for creating a new speaker group. In scenarios where an Accused Google Player is added to a new speaker group being created, the group begins in an uninvoked state (or an unlaunched state in Google's terms) – not a "state of stopped playback" as Dr. Schonfeld contends – and the Accused Google Player makes no reference to the "group's state" when handling the "join_group" message indicating that the Accused Google Player has been added to the new speaker group.

101. Indeed, an Accused Google Player installed with newly-released firmware version 1.56.324896 carries out the same functionality for handling the "join_group" message that was previously carried out by Accused Google Players installed with prior firmware versions, which undisputedly did not involve any "match[ing]" of the "group's state," along with one additional call to the "StopCurrentApp()" function discussed above. However, this "StopCurrentApp()" function does not make any reference to the "group's state," let alone causes the Accused Google Player to "match the group's state of stopped playback" as Dr. Schonfeld contends. Instead, the "StopCurrentApp()" function merely causes the Accused Google Player to stop its current receiver app, to the extent that such a receiver app is running. In this respect, if the receiver app being run by the Accused Google Player is currently causing the Accused Google Player to engage in active playback, then the "StopCurrentApp()" function will cause the Accused Google Player to stop that active playback, whereas if the receiver app being run by the Accused Google Player is not currently causing the Accused Google Player to engage in active playback, then the "StopCurrentApp()" function will not impact the playback state of the Accused Google Player – but in either case, the additional call to the "StopCurrentApp()" function does not cause an Accused Google Player to "match the group's state of stopped playback." And in a scenario where the Accused Google Player is not currently running a receiver app, the "StopCurrentApp()" function will have no impact at all on the state of the Accused Google Player. The foregoing operation of the "StopCurrentApp()" function is confirmed by the testimony of Mr. MacKay cited above.

102. Turning to Dr. Schonfeld's discussion of the functionality for modifying a pre-existing speaker group that is encoded within newly-released firmware version 1.56.324896, Dr. Schonfeld begins that discussion by making the following statement:

added to a pre-existing speaker group that is in an *uninvoked state*. In those scenarios, the Accused Google Player makes no reference to the "behavior of the group" while handling the "join_group" message indicating that the Accused Google Player has been added to the pre-existing group, and if the Accused Google Player was operating in standalone mode prior to being added to a pre-existing group in an unlaunched state, the Accused Google Player will continue to operate in standalone mode after being added to the pre-existing group. This is confirmed by the testimony of Mr. MacKay cited above.

108. Further, while I agree that an Accused Google Player installed with newly-released firmware version 1.56.324896 "*does not* play back music as a member of the group" in any scenario where the Accused Google Player is being added to a pre-existing speaker group that is in an *uninvoked state*, I disagree with Dr. Schonfeld's suggestion that this functionality is premised on "the group's behavior beforehand." Again, the Accused Google Player makes no reference to the "the group's behavior" while handling the "join_group" message indicating that the Accused Google Player has been added to the pre-existing group, as confirmed by the testimony of Mr. MacKay cited above.

109. Further yet, I disagree with Dr. Schonfeld's suggestion that "the speaker joined to the group switches to group playback upon being joined to the group." In any scenario where a pre-existing speaker group is in an *uninvoked state* when an Accused Google Player installed with newly-released firmware version 1.56.324896 is added to the speaker group, the Accused Google Player will continue to operate in standalone mode after being added to the pre-existing group rather than "switch[ing] to group playback," as confirmed by the testing I observed, the source code for newly-released firmware version 1.56.324896, and Mr. MacKay's testimony. In fact, Dr. Schonfeld never once even suggests that adding an Accused Google Player installed with newly-released firmware version 1.56.324896 to a pre-existing speaker group in an *uninvoked state* would cause the pre-existing speaker group to become *invoked*, which is what dictates whether the group members are configured for "grouped playback" in accordance with the group. *See, e.g.*, 1/25/2023 K. MacKay Rough Dep. Tr. at 34:10-37:23 (Google's corporate designee describing the distinction between a launched speaker group and an unlaunched speaker group and confirming

that (i) a follower begins "taking part in group playback" once it "receives a launch command from the group leader" and "launches the multizone follower app," (ii) the leader begins "taking part in group playback" once "the app has finished launching" on the leader device, and (iii) "if a group is in the unlaunched state, then the devices that are members of that group would not be playing as part of that group"). Thus I fail to see what basis Dr. Schonfeld has for saying that the Accused Google Player being added "switches to grouped playback upon being joined to the group" in these scenarios where the pre-existing speaker group would remain in an *uninvoked state* after the Accused Google Player is added.

110. Turning next to Dr. Schonfeld's discussion of the functionality for creating a new speaker group that is encoded within newly-released firmware version 1.56.324896, Dr. Schonfeld begins that discussion by making the following statement:

> I now describe the behavior of a speaker added to a group where the group was not previously created. As shown below, regardless of whether one speaker being added to the new group is playing music, whether multiple speakers being added to the new group are playing music, whether speakers being added to the new group are playing the same music or different music, or whether no speakers being added to the new group are playing music, the result is the same: In each instance, each speaker added to the new group acts as a member of the group by not playing back any music. Further, each speaker added to the new group leaves its prior playback state, and its playback is stopped at the same time and in conjunction with every other speaker in the new group. Below, I provide various examples of this behavior and I discuss the relevant source code enabling this functionality. Each scenario begins with no group previously created.

Schonfeld Rebuttal Report at ¶ 54. However, this statement is flawed for several reasons.

111. First, Dr. Schonfeld's suggestion that "each speaker added to the new group acts as a member of the group by not playing back any music" is incomplete, misleading, and inaccurate. While an Accused Google Player installed with newly-released firmware version 1.56.324896 that is added to a new speaker group will internally memorialize that it has been added as a member of the new speaker group in the same way that Accused Google Players installed with prior firmware versions would have done, an Accused Google Player installed with newly-released firmware version 1.56.324896 does not "act[] as a member of the group" in terms of its playback behavior, as Dr. Schonfeld appears to suggest. To the contrary, an Accused Google Player that is operating

in standalone mode before being added to a new speaker group will continue to operate in standalone mode after being added to the new speaker group, as confirmed by the testing I observed, the source code for newly-released firmware version 1.56.324896, and Mr. MacKay's testimony. In this respect, the fact that an Accused Google Player installed with newly-released firmware version 1.56.324896 does not engage in active playback after being added to a new speaker group is due to the Accused Google Player's additional call to the "StopCurrentApp()" function, which has nothing to do with the playback behavior of the new speaker group. I further note that, because the new speaker group did not previously exist and is in an uninvoked state at the time of creation, it is not clear to me what playback behavior Dr. Schonfeld is even referring to.

112. Second, I disagree with Dr. Schonfeld's suggestion that "each speaker added to the new group *leaves its prior playback state*, and its playback is stopped at the same time and in conjunction with every other speaker in the new group." As confirmed by the testing I observed, which is summarized above in Section IX.A, as well as the source code for newly-released firmware version 1.56.324896, which is summarized above in Section IX.B, an Accused Google Player installed with newly-released firmware version 1.56.324896 that is operating in standalone mode and not engaging in active playback at the time when it is added to a new speaker group will remain in its "prior playback state" by continuing to operate in standalone mode and continuing to not engage in active playback after being added to the new speaker group. Moreover, while an Accused Google Player installed with newly-released firmware version 1.56.324896 that is operating in standalone mode and is individually engaging in active playback at the time when it is added to a new speaker group will stop engaging in active playback after being added to the new speaker group, this functionality is due to the Accused Google Player's additional call to the "StopCurrentApp()" function, which has nothing to do with the playback behavior of the new speaker group.

113. For similar reasons, I also disagree with Dr. Schonfeld's characterization of the scenarios he shows and describes at paragraphs 55-57 of his Rebuttal Report. *See* Schonfeld Rebuttal Report at ¶¶ 55-57. In those paragraphs, Dr. Schonfeld shows scenarios where new

speaker groups of Accused Google Players installed with newly-released firmware version 1.56.324896 are created, and Dr. Schonfeld describes the end result of those scenarios in terms of the Accused Google Players "operat[ing] as a group playing back no music," which is not an accurate characterization. What Dr. Schonfeld fails to acknowledge or recognize is that in each scenario in which a new speaker group is created, the new speaker group starts out in an *uninvoked state* (or an unlaunched state in Google's terms), which means that the Accused Google Players added to the new speaker group are not "operat[ing] as a group" as a result of the new speaker group being created. To the contrary, the Accused Google Players added to the new speaker group do not begin "operat[ing] as a group" until the new speaker group is later invoked at the request of a user, and as such, any Accused Google Players that is operating in standalone mode at the time that it is added to a new speaker group will continue to operate in standalone mode after being added to the new speaker group, rather than transitioning to a grouped mode. *See, e.g.*, 1/25/2023 K. MacKay Rough Dep. Tr. at 34:10-37:23 (Google's corporate designee describing the distinction between a launched speaker group and an unlaunched speaker group and confirming that (i) a follower begins "taking part in group playback" once it "receives a launch command from the group leader" and "launches the multizone follower app," (ii) the leader begins "taking part in group playback" once "the app has finished launching" on the leader device, and (iii) "if a group is in the unlaunched state, then the devices that are members of that group would not be playing as part of that group").

114. Dr. Schonfeld concludes his discussion of the functionality for creating a new speaker group that is encoded within newly-released firmware version 1.56.324896 as follows:

> As shown above, in every instance, the speakers added to the group act in conjunction with the group once the new group is created. The speakers leave their prior state, stop playback in unison, and remain stopped in conjunction with the group.

Schonfeld Rebuttal Report at ¶ 53. However, I disagree with this conclusion for many of the same reasons just discussed.

115. For instance, to the extent Dr. Schonfeld's is using the phrase "act in conjunction with the group" to mean operate in accordance with the group, I disagree with Dr. Schonfeld's

may do this in a responsive report or in a supplemental report as appropriate.

299. I expect to testify at trial regarding the matters set forth in this report, if asked about these matters by the Court or by the parties' attorneys.

Dated: January 26, 2023

By: 
Kevin C. Almeroth